Unit 2:

- Matrix
- Loop(while, repeat)
- Function

Matrix

- Matrix: set of objects indexed by rows and columns

```
matrix(data,nrow,ncol,byrow, …)
```

- To create a matrix:
  - data: a vector with the data or elements. These elements have to be of the same data type (numeric, character, logic).
  - nrow: number of rows
  - ncol: number of columns
  - byrow: logical. If FALSE (the default) the matrix is filled by columns, otherwise the matrix is filled by rows.

Matrix

matrix(1:6, nrow=2)	matrix(1:6,nrow=3)	matrix(1:6, nrow=2,byrow=T)
[,1] [,2] [,3] [1,] 1 3 5 [2,] 2 4 6	[,1] [,2] [1,] 1 4 [2,] 2 5 [3,] 3 6	[,1] [,2] [,3] [1,] 1 2 3 [2,] 4 5 6

Matrix

- To select elements of a matrix, you have to use: row and column

“[“ + row+ “, “+ column + ”]”

```
>x=matrix(1:6, nrow=3)
          [,1] [,2]
     [1,]   1    4
     [2,]   2    5
     [3,]   3    6
```

>x[1,] [1] 1 4	#First row
> x[,1] [1] 1 2 3	#First column
>x[2,2] [1] 5	#Select the element of second row and column

Matrix: Functions

```
>x=matrix(1:6, nrow=3)
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
```

```
>cbind(x,c(10,10,10))

     [,1] [,2] [,3]
[1,]    1    4   10
[2,]    2    5   10
[3,]    3    6   10
```

Add column

```
> rbind(x,c(20,20))

     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
[4,]   20   20
```

Add row

Matrix: Functions

```
> mat=matrix(c(20,65,1.74,22,70,1.80,19,68,1.70),nrow=3,byrow=T)
> mat
      [,1] [,2]  [,3]
[1,]   20   65 1.74
[2,]   22   70 1.80
[3,]   19   68 1.70
```

```
>colnames(mat)=c("edad","peso","altura")
#Name columns
> mat
     edad peso altura
[1,]   20   65   1.74
[2,]   22   70   1.80
[3,]   19   68   1.70
```

```
>rownames(mat)=c("juan","maria","ana")
#Name rows
> mat
      edad peso altura
juan    20   65   1.74
maria   22   70   1.80
ana     19   68   1.70
```

Matrix: Functions

```
>mat
                 edad peso altura
      juan        20     65   1.74
      maria       22     70   1.80
      ana         19     68   1.70
```

>length(mat) [1] 9	# number of elements of matrix
>mode(mat) [1] "numeric "	# data type of matrix
>dim(mat) [1] 3 3	# dimensions of the matrix
>dimnames(mat) [[1]] [1] "juan" "maria" "ana" [[2]] [1] "edad" "peso" "altura"	# names of dimensions of the matrix

Matrix: Functions

```
>mat
                edad   peso  altura
      juan       20     65      1.74
      maria      22     70      1.80
      ana        19     68      1.70
```

> rownames(mat) [1] "juan" "maria" "ana"	#name of rows
> colnames(mat) [1] "edad" "peso" "altura"	#name of columns
>is.matrix(x) [1] TRUE	# is x a matrix?

Matrix: Functions

- Also, you can use rows and columns name to select elements of the matrix

```
> mat
         edad peso altura
juan      20   65   1.74
maria     22   70   1.80
ana       19   68   1.70
```

```
>mat["juan", ]

  edad   peso altura
 20.00  65.00   1.74
```

```
>mat [,"edad"]

 juan maria   ana
   20     22     19
```

```
>mat[ , c("edad","altura")]

          edad altura
juan      20   1.74
maria     22   1.80
ana       19   1.70
```

```
> dimnames(mat)
[[1]]
[1] "juan"  "maria" "ana"

[[2]]
[1] "edad"   "peso"   "altura "
```

```
> apply(mat,2,mean)
     edad          peso           altura
20.333333   67.666667   1.746667
```

Matrix: Functions

- Example: dimnames () to name rows and columns

```
> dimnames(mat)<-list (NULL, paste("Student-",1:3, sep=""))
> mat
    Student-1 Student-2 Student-3
[1,]        20        65      1.74
[2,]        22        70      1.80
[3,]        19        68      1.70
```

```
> mean(mat[3,])
[1] 29.56667
```

```
> apply(mat,2,mean)
Student-1  Student-2   Student-3
20.333333 67.666667  1.746667
```

Loop: while and repeat

Loop: while()

- Syntax:

```
#1.-Initialize a variable
while (logic condition) #2.- Check a logic condition
{
    sentences to execute
   #3.- Modify variable
}
```

- Example:

```
i=0
while (i<5)
{
    i=i+1
   print(i)
}
```

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

Loop: repeat()

- Syntax :

```
#Initialize a variable
repeat {
      sentences to execute
     # Modify variable
     # Check an exit condition

}
```

- Example:

```
i=0
 repeat
 {
      i=i+1
     print(i)
     if (i>=5)
       break
}
```

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

Functions

More about functions

- You can enter specific values to use in functions throught parameters in the definition of the functions

```
Nombre_func <- function(arg_1,arg_2,…,arg_n)
{
   sentences to execute
}
```

- The call to functions with parameters is:

```
Nombre_func (expr_1, expr_2,…,expr_n)
```

More about functions

- Example:

Definition of the function:	**> sumGrades<-function(a,b)**
	{
	result=a+b
	return(result)
	}

Call to the function	> sumGrades(3,3) [1] 6

More about functions

- You can write a default value to parameters. In this case, when you call to the function, you have to write values to parameters which not have a default value.

```
> sumgrades<-function(a=2,b)
 {
    result=a+b
    return(result)
 }
```

```
> sumgrades(,3)
[1] 5
```

More about functions

- It allows you to place arguments in the order you want, as long as you specify which argument corresponds to.

```
> divGrades<-function(a,b)
 {
      result=a/b
      return(result)
 }
```

```
> divGrades(4,2)
[1] 2
```

```
> divGrades(2,4)
[1] 0.5
```

```
> divGrades(b=2,a=4)
[1] 2
```